Kapitola: Gramatiky a prepisovacie systémy

1.1 Frázové gramatiky

Viz. skriptá z foje.

1.2 Markov algorithm

http://en.wikipedia.org/wiki/Markov_algorithm

Deterministická verzia frázových gramatík.

Markovov algoritmus je trojica (Σ, Γ, P), kde Σ je vstupná, $\Gamma \supseteq \Sigma$ je pracovná abeceda a P je konečná postupnosť prvkov z $\Gamma^* \times \Gamma^* \times \{0, 1\}$.

Prvky P voláme pravidlá, pravidlo $(u, v, 0)$ zapisujeme ako $(u \to v)$ a pravidlo $(u, v, 1)$ ako $(u \to \bullet v)$. Pravidlá tvaru $(u, v, 1)$ voláme terminálne.

Konfigurácia je slovo nad Γ^*. Krok výpočtu zo slova w vyzerá nasledovne:

1. Ak sa v w nenachádza žiadna ľavá strana pravidla, výpočet končí.
2. Nájdeme prvé pravidlo, ktorého ľavá strana sa nachádza niekde v w.
3. Ak je výskytov viac, vyberieme najľavejší z nich.
4. Vybraný výskyt nahradíme pravou stranou zodpovedajúceho pravidla.
5. Ak išlo o terminálne pravidlo, výpočet končí.

(**Pozor**, kroky 2 a 3 sú v opačnom poradí oproti verzii prezentovanej na prednáške. Takto je to pohodlnejšie, ľahšie sa dokáže univerzálnosť.)

Príklad: Markov algorithm na preklad čísla z binárnej do unárnej sústavy.

$\Sigma = \{0, 1, x\}$, $P = \{\ x0 \to 0xx, 1 \to 0x, 0 \to \varepsilon\ \}$.

Výpočet na reťazci 1010:

```
1010 => 0x010 => 00xx10 => 00xx0x0 => 00x0xxx0 => 000xxxxx0 => 000xxxx0xx => 000xxx0xxxx => 000xx0xxxxxx => 000x0xxxxxxxx =>
=> 0000xxxxxxxxxx => 000xxxxxxxxxx => 00xxxxxxxxxx => 0xxxxxxxxxx => xxxxxxxxxx
```

1.2.1 Univerzalita

Vieme simulovať DTS – budeme postupne vyrábať jeho konfigurácie.

Uvažujme normálny tvar DTS, ktorý sa nikdy nezasekne, t. j. buď v konečnom čase akceptuje alebo beží do nekonečna. (Ľubovoľný DTS ľahko upravíme na takýto doplnením jeho prechodovej funkcie.)

Teraz jednoducho pre každé pravidlo prechodovej funkcie spravíme sadu prepisovacích pravidiel. Pre $(q, y, 1) = \delta(p, x)$ budeme mať pravidlo $px \to yq$, pre $(q, y, -1) = \delta(p, x)$ budeme mať pravidlá $zpx \to qzy$ pre každé z. Tieto pravidlá môžu byť usporiadané ľubovoľne – vždy sa bude dať použiť len jedno z nich. Tie z nich, ktoré vedú do akceptačného stavu, prehlásime za terminálne.

Na záver pridáme pravidlo $\varepsilon \to q_0$. Toto sa použije len vtedy, ak nevieme použiť žiadne z predchádzajúcich – teda ak vetná forma ešte neobsahuje symbol pre stav. Vtedy sa teda použije toto pravidlo, a to na najľavejšom mieste kde sa môže – na začiatku vetnej formy.

1.3 Postov tag systém

Tag systém je trojica (m, Σ, P), kde $m > 0$ je celé číslo, Σ je konečná abeceda a $P : \Sigma \to \Sigma^*$ sú prepisovacie pravidlá. Konfigurácia je slovo nad abecedou Σ. Krok výpočtu na slove w je definovaný nasledovne: ak $|w| < m$, zaseknem sa, inak prejdem do konfigurácie $w_{m+1} \ \ldots w_{|w|} P(w_1)$.

1.3.1 Príklad tag systému

$$T = (\ 2,\ \{a, b, c\},\ \{a \to bc, b \to a, c \to aaa\}\ )$$

Tento tag systém počíta jemnú modifikáciu tzv. Collatz sequence http://en.wikipedia.org/wiki/Collatz_conjecture: ak začneme zo slova a^k, tak slová tvaru a^x pre $x > 0$, ktoré uvidíme počas výpočtu, predstavujú modifikovanú Collatz sequence pre k.

Collatz sequence: ak je n párne, je jeho nasledovník $n/2$, ak je $n = 2k + 1$ pre $k > 0$, jeho nasledovník je $3n + 1$, ak je $n = 1$, postupnosť končí.

Môžeme si všimnúť, že pre nepárne n je nasledujúci člen, $3n + 1$, vždy párny, a teda za ním nasleduje $(3n + 1)/2$. Náš tag systém preskočí hodnotu $3n + 1$, teda v ňom po n bude nasledovať priamo $(3n + 1)/2$.

```
Computation for the word "aaa":
aaa <--> 3
  abc
    cbc
      caaa
        aaaaa  <--> 5
          aaabc
            abcbc
              cbcbc
                cbcaaa
                  caaaaaa
                    aaaaaaaa  <--> 8
                      aaaaaabc
                        aaaabcbc
                          aabcbcbc
                            bcbcbcbc
                              bcbcbca
                                bcbcaa
                                  bcaaa
                                    aaaa  <--> 4
                                      aabc
                                        bcbc
                                          bca
                                            aa  <--> 2
                                              bc
                                                a  <--> 1
```

Collatz conjecture: každá táto postupnosť končí jednotkou. Ekvivalentná formulácia: pre každý vstup tvaru a^k sa náš tag systém časom zasekne.

Aktuálny stav: v 2008 sa vie, že Collatz conjecture platí zhruba po 5.48×10^{18}, ďalej ništ. Inými slovami, o tomto konkrétnom maličkom tag systéme sa nevie, či vždy zastane.

Jeden rekord pre ilustráciu: pre $n = 1\,980\,976\,057\,694\,848\,447 \sim 2 \times 10^{18}$ má Collatz sequence ešte 622 členov, najväčší z nich je $32\,012\,333\,661\,096\,566\,765\,082\,938\,647\,132\,369\,010 \sim 3.2 \times 10^{37}$ (po 399 krokoch). Jeho hodnota (zvaná maximum excursion) je približne 8.15753-násobok n^2. Viac na http://www.ieeta.pt/~tos/3x+1.html

1.3.2 2-tag halting problem

Vstup: kladné celé n a postupnosť $n + 1$ slov $P_1, \ldots, P_n, Q$ nad abecedou $\{1, \ldots, n\}$.
Otázka: Začneme z Q a dokola opakujeme: ak máš slovo xyW, zmeň ho na WP_x. Je tento proces konečný?

Toto je asi najjednoduchšie popísateľná verzia halting problému.

1.3.3 Tag systémy sú Turing complete

Viz. Minsky, M. L. "Recursive Unsolvability of Post's Problem of 'tag' and Other Topics in Theory of Turing Machines." Ann. of Math. 74, 437-455, 1961, http://www.wolframscience.com/prizes/tm23/images/Minsky.pdf

Myšlienka: Minsky ukáže, že registrový stroj s dvoma countrami je Turing complete. Konštrukciu sme už videli: v jednom counteri je v mocninách prvočísel zakódovaných niekoľko iných, druhý slúži na pomocné výpočty. Minský následne zostrojí ekvivalentný tag systém, ktorého slovo vo vhodných okamihoch má tvar $Q_j a_j^n$, kde Q_j je stav simulovaného automatu, a_j je jemu zodpovedajúci pomocný symbol a n je hodnota v prvom counteri.